

# Jenkins 를 이용한 빌드 자동화와 배포

작성자 : 강경미([carami@nate.com](mailto:carami@nate.com))

root 사용자로 해야할 일

- 일반 계정 생성
- 일반 계정이 접근할 수 있는 디렉토리 생성 및 권한 주기(tomcat, jenkins 등 설치 폴더)
- JDK & Maven 설치 및 환경설정
- 포트포워딩(80 에서 tomcat 운영 포트)

일반 계정으로 해야할 일

- Jenkins 설치 및 실행
- mysql 에 사용자 계정 생성 및 데이터 베이스 생성
- tomcat 설치 및 실행
- Jenkins 설정

## 1. 사용자 계정 생성

서버에 root 계정으로 접속하여 사용자 계정을 생성한다.

- 조에서 함께 사용할 계정을 하나 만든다. adduser 명령을 통해 만들 수 있다.
- 개인 계정을 만든다.
- 접속 후 암호변경은 passwd 명령을 통해 변경할 수 있다. 대문자, 소문자, 특수문자등을 잘 섞어서 만든다. 1234 이런 암호는 해킹당함.

adduser 아이디
-------------

## 2. 일반 계정이 접근할 수 있는 디렉토리 생성 및 권한 주기(tomcat, jenkins 등 설치 폴더)

root 계정으로 application 이 설치될 디렉토리를 생성한 후, 해당 사용자에게 권한을 준다.

- 디렉토리를 생성한다. mkdir 명령

- 조에서 함께 사용할 아이디에게 권한을 준다. chown 명령.
- 목록을 조회한다. ls 명령

```
mkdir /apps
chown 아이디.아이디 /apps
ls -la
```

### 3. JDK & Maven 설치

#### 3-1. JDK 설치

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

- jdk-8u141-linux-x64.tar.gz 를 다운로드한다.

ftp 를 이용하여 업로드 한다. windows 사용자는 <https://filezilla-project.org/> 를 사용한다.

맥의 경우는 다음과 같이 접속하여 업로드 할 수 있다.

```
sftp 아이디@ip 주소
sftp> bin
sftp> hash
sftp> jdk-8u141-linux-x64.tar.gz
```

업로드 이후 다음과 같이 jdk 를 설치한다.

- '~/ ' 는 홈 디렉토리를 말한다. '/home/아이디' 와 같다.
- /apps 디렉토리로 이동한 후, 홈디렉토리에 업로드한 파일을 복사해온다. cp 는 복사명령
- apps 폴더에서 압축을 해제한다. xvfz 옵션은 gz 압축을 해제한 후 tar 를 풀라는 의미를 가진다.
- 압축이 풀린 jdk1.8.0\_141 폴더를 jdk 로 심볼릭링크를 건다. /apps/jdk 는 JAVA\_HOME 경로로 지정할 예정이다.
- ls -la 명령으로 확인한다.

```
cd /apps
cp ~/jdk-8u141-linux-x64.tar.gz .
```

```
tar xvfz jdk-8u141-linux-x64.tar.gz
ln -s jdk1.8.0_141/ jdk
ls -la
```

### 3-2. maven 설치

```
cd /apps
wget http://mirror.navercorp.com/apache/maven/maven-3/3.5.0/binaries/apache-maven-3.5.0-bin.tar.gz
tar xvfz apache-maven-3.5.0-bin.tar.gz
ln -s apache-maven-3.5.0 maven
```

### 3-3. JDK & maven 환경설정

JAVA\_HOME, CLASSPATH, PATH 환경설정을 추가한다. root 계정으로 수정해야한다.

'su -' 명령은 root 사용자로 전환하는 것이다.

```
su -
nano /etc/profile
```

Nano 에디터 명령

저장 : ctrl+o 후에 엔터

나가기 : ctrl+x 후에 엔터

profile 의 맨 아래에 다음을 추가한다.

```
JAVA_HOME=/apps/jdk
CLASSPATH=.:$JAVA_HOME/lib/tools.jar

PATH=$PATH:$HOME/bin:$JAVA_HOME/bin

export PATH CLASSPATH JAVA_HOME
```

수정된 내용을 적용하려면 다음과 같이 명령을 내린다. 혹은 재접속한다.

```
source /etc/profile
```

다음과 같이 jdk 가 잘 설치되었는지 확인한다.

```
java -version  
  
echo $JAVA_HOME  
echo $PATH  
echo $CLASSPATH
```

#### 4. Jenkins 설치

##### 4-1. 설치

조에서 함께 사용할 아이디(필자는 carami 란 아이디를 사용)로 접속, Jenkins 설치

- jenkins 를 다운로드 한다. Generic Java package (.war) 를 다운로드 한다.
- jenkins 는 독립적으로도 실행가능하다.
- java -jar 명령으로 실행할 수 있다. '&'는 리눅스에서 백그라운드로 실행하라는 명령이다. 로그아웃해도 실행되고 있다.

```
cd /apps  
wget http://mirrors.jenkins.io/war-stable/latest/jenkins.war  
java -jar jenkins.war &
```

##### 4-2. 젠킨스 실행

```
http://ip:8080
```

으로 접속할 수 있다.

- /home/계정/.jenkins/secrets/initialAdminPassword 에 있는 암호를 입력해야 한다.
- cat 명령은 파일의 내용을 보는 명령

```
cat /home/계정/.jenkins/secrets/initialAdminPassword
```

Getting Started


# Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

`/Users/urstory/.jenkins/secrets/initialAdminPassword`

Please copy the password from either location and paste it below.

Administrator password



Continue

Getting Started

# Customize Jenkins


Plugins extend Jenkins with additional features to support many different needs.

## Install suggested plugins

Install plugins the Jenkins community finds most useful.

## Select plugins to install

Select and install plugins most suitable for your needs.



Jenkins 2.60.2

Getting Started

# Getting Started

<input type="checkbox"/> Folders Plugin	<input type="checkbox"/> OWASP Markup Formatter Plugin	<input type="checkbox"/> build timeout plugin	<input type="checkbox"/> Credentials Binding Plugin	<b>** recommended API plugin</b>
<input type="checkbox"/> Timestamp	<input type="checkbox"/> Workspace Cleanup Plugin	<input type="checkbox"/> Ant Plugin	<input type="checkbox"/> Gradle Plugin	
<input type="checkbox"/> Pipeline	<input type="checkbox"/> GitHub Branch Source Plugin	<input type="checkbox"/> Pipeline: GitHub Groovy Libraries	<input type="checkbox"/> Pipeline: Stage View Plugin	
<input type="checkbox"/> Git plugin	<input type="checkbox"/> Subversion Plugin	<input type="checkbox"/> SSH Slaves plugin	<input type="checkbox"/> Matrix Authorization Strategy Plugin	
<input type="checkbox"/> PAM Authentication plugin	<input type="checkbox"/> LDAP Plugin	<input type="checkbox"/> Email Extension Plugin	<input type="checkbox"/> Mako Plugin	
				<b>** -- required dependency</b>

Jenkins 2.60.2

Getting Started

## Create First Admin User

계정명:

암호:

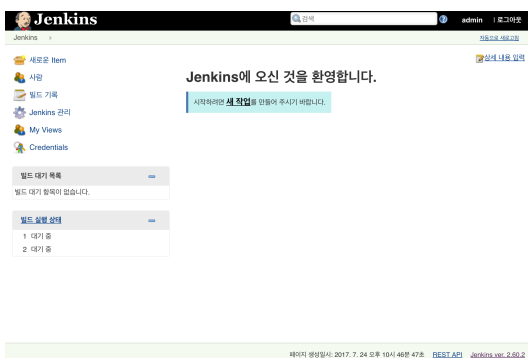
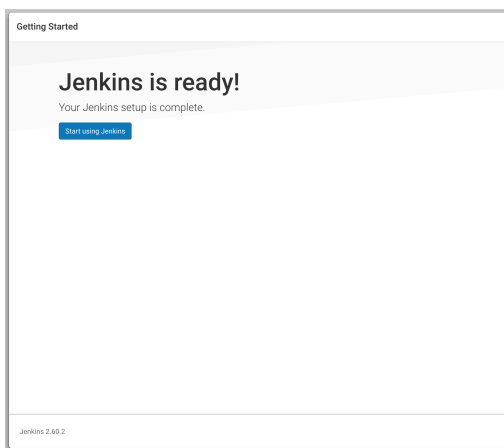
암호 확인:

이름:

이메일 주소:

Jenkins 2.60.2 [Continue as admin](#) [Save and Finish](#)

save and finish 버튼을 클릭한다.



#### 4-3. 젠킨스 종료

- ps 명령을 프로세스 목록을 보는 명령
- '|'는 파이프 라고 읽는다. ps 의 결과를 파이프 기호 뒷편으로 전달하는 목적으로 사용된다.
- 표준 출력을 우측프로그램의 표준입력으로 제공

- grep 명령은 입력받은 내용중에서 특정 문자열이 포함된 것을 찾는다.

```
ps -lef | grep jenkins
```

```
[carami@boostcamp-001 ~]$ ps -lef | grep jenkins
0 S carami    1427    871   6   80    0 - 903049 futex_ 16:23 pts/4    00:00:59 java -jar
jenkins.war
0 S carami    2551   2530   0   80    0 - 25814 pipe_w 16:39 pts/6    00:00:00 grep
jenkins
```

위와 같이 나올경우 1427, 2551 이 프로세스 id 이다. 1427 을 종료시킨다.

```
kill -9 1427
```

## 5. mysql 에 사용자 계정 생성 및 데이터 베이스 생성

- mysql 에 접속한다.

```
mysql -uroot mysql
```

암호를 대문자,소문자,특수문자, 숫자등으로 잘 만들지 않을 경우 다음과 같은 오류가 발생한다.

Your password does not satisfy the current policy requirements

```
create user 'carami'@localhost identified by '암호';
create user 'carami'@'%' identified by '암호';

create database tododb;

GRANT ALL on tododb.* TO 'carami'@'localhost';
GRANT ALL on tododb.* TO 'carami'@'%';

flush privileges;
```

## 6. tomcat 설치

```
cd /apps
wget http://apache.mirror.cdnetworks.com/tomcat/tomcat-8/v8.5.16/bin/apache-tomcat-
```

```
8.5.16.tar.gz
tar xvfz apache-tomcat-8.5.16.tar.gz
ln -s apache-tomcat-8.5.16 tomcat
```

- jenkins 배포를 위한 tomcat 설정 파일 수정

tomcat 의 실행 포트를 수정하기 위해 tomcat/conf/server.xml 파일을 수정한다.

nano server.xml

```
<Connector port="8080" protocol="HTTP/1.1"
            connectionTimeout="20000"
            redirectPort="8443" />
```

위와 같은 부분을 찾아 다음과 같이 수정한다.

```
<Connector port="9090" protocol="HTTP/1.1"
            connectionTimeout="20000"
            redirectPort="8443" URIEncoding="UTF-8"/>
```

- webapps 폴더에 있는 기본 웹 어플리케이션들을 삭제한다. 빌드한 웹 어플리케이션을 jenkins 에서 ROOT.war 파일로 복사하여 설치할 예정이다.

```
cd /apps/tomcat/webapps
rm -rf *
ls -la
```

- tomcat 의 실행

현재 경로의 파일을 실행하려면 리눅스에서는 './' 를 앞에 써줘야 한다.

```
cd /apps/tomcat/bin
./startup.sh
```

- tomcat 의 종료



```
cd /apps/tomcat/bin
./shutdown.sh
```

## 7. 80 -> 9090 으로 포트포워딩

참고문서 : <https://blog.outsider.ne.kr/580>

1024 이하로 동작하는 어플리케이션을 실행하기 위해서는 root 권한이 필요하다. tomcat 을 일반 사용자 계정으로 실행하기 위하여 9090 포트로 설정하였다. 그런데 <http://ip> 로 웹 어플리케이션이 호출되기 위해서는 80 포트로 동작해야한다. 이런 문제를 해결하기 위해서 root 사용자로 포트포워딩 설정을 한다. 80 으로 오는 요청을 9090 으로 전달하는 것이다.

아래의 명령은 root 권한으로 실행해야 한다.

```
iptables -A PREROUTING -t nat -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 8080
```

확인은 다음 명령을 이용한다.

```
iptables -t nat -L
```

## 8. jenkins 를 이용하여 빌드 및 배포하기

참고문서 : <http://handcoding.tistory.com/25>

<http://jojoldu.tistory.com/139>

먼저 nano /apps/tomcat/deploy.sh 명령을 이용하여 shell 스크립트를 작성한다.

/apps/tomcat/bin 으로 시작하는 tomcat 프로세스가 없다면, /home/carami/.jenkins/workspace/todo/target/todo-0.0.1-SNAPSHOT.war 파일을 /apps/tomcat/webapps/ROOT.war 로 복사한다. (아직 todo-0.0.1-SNAPSHOT.war 파일이 존재하지 않다.)

그리고 나서 /apps/tomcat/bin/startup.sh 을 실행하여 톰캣을 실행한다. BUILD\_ID=dontKillMe 에 대한 설명은 <http://lmg1982.tistory.com/178> 를 참고.

이미 tomcat 이 실행중이라면 tomcat 프로세스를 종료한 후, 위의 작업을 진행한다. /home/carami 경로를 본인에 맞게 알맞게 수정한다.

```
#!/bin/sh

if [ -z "`ps -eaf | grep java|grep /apps/tomcat/bin`" ]; then
    cp      /home/carami/jenkins/workspace/todo/target/todo-0.0.1-SNAPSHOT.war
/apps/tomcat/webapps/ROOT.war
    BUILD_ID=dontKillMe /apps/tomcat/bin/startup.sh
else
    ps -eaf | grep java | grep /apps/tomcat/bin | awk '{print $2}' |
    while read PID
    do
        echo "Killing $PID ..."
        kill -9 $PID
        echo
        echo "Tomcat  is being shutdownned."
    done
    cp      /home/carami/jenkins/workspace/todo/target/todo-0.0.1-SNAPSHOT.war
/apps/tomcat/webapps/ROOT.war
    BUILD_ID=dontKillMe /apps/tomcat/bin/startup.sh
fi
```

해당 스크립트가 실행될 수 있도록 퍼미션을 설정한다.

- chmod 는 퍼미션을 변경하는 명령이다. 자세한 내용은 찾아보세요.

```
chmod 755 deploy.sh
```

- Jenkins 에서 Item 추가하기



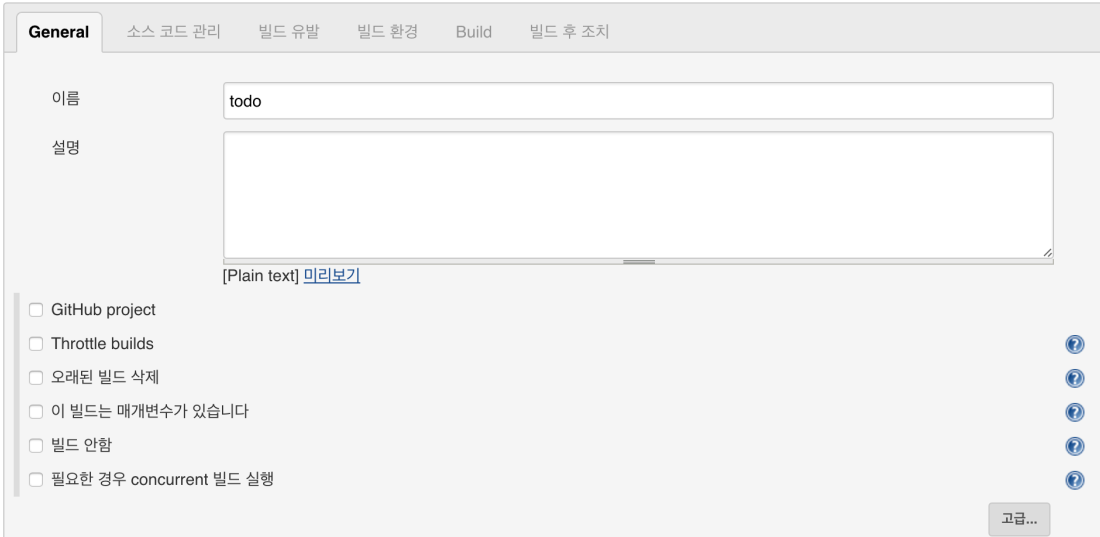
새로운 Item 을 클릭한다.



The image shows a Jenkins dialog box titled "Enter an item name". It contains a text input field that is currently empty. Below the input field, there is a red error message: "» This field cannot be empty, please enter a valid name". Below the error message, there is a section titled "Freestyle project" with a small icon of a box and a lightbulb. The text below the icon says: "이것은 Jenkins의 주요 기능입니다. Jenkins은 어느 빌드 시스템과 어떤 SCM(형상관리)으로 묶인 당신의 프로젝트를 빌드할 것이고, 소프트웨어 빌드보다 다른 어떤 것에 자주 사용될 수 있습니다."

item name 에 프로젝트 이름을 적는다. 필자는 todo 라고 입력함. 그리고 나서 아래에 있는 Freestyle project 를 클릭한 후 하단의 OK 버튼을 클릭한다.

아래의 이미지와 같이 설정한다.



The image shows the Jenkins "General" configuration page for a Freestyle project. The page has a tabbed interface with "General" selected. The "이름" (Name) field is set to "todo". The "설명" (Description) field is empty. Below the description field, there is a link that says "[Plain text] [미리보기](#)". On the left side, there are several checkboxes: "GitHub project", "Throttle builds", "오래된 빌드 삭제" (Delete old builds), "이 빌드는 매개변수가 있습니다" (This build has parameters), "빌드 안함" (Do not build), and "필요한 경우 concurrent 빌드 실행" (Run concurrent builds if needed). On the right side, there are five question mark icons. At the bottom right, there is a button labeled "고급..." (Advanced...).

### 소스 코드 관리

☐ None  
☒ Git

Repositories

Repository URL 
  
Credentials

Branches to build

Branch Specifier (blank for 'any')

Repository browser

Additional Behaviours

☐ Subversion

github URL 을 적고 하단의 Add 버튼을 클릭하여 github id/password 를 입력한다.  
checkout 할 branch 를 적는다. 여기에서는 master 를 적었다.

### 빌드 유발

☐ 빌드를 원격으로 유발 (예: 스크립트 사용)  
☐ Build after other projects are built  
☐ Build periodically  
☐ GitHub hook trigger for GITScm polling  
☐ Poll SCM

### 빌드 환경

☐ Delete workspace before build starts  
☐ Abort the build if it's stuck  
☐ Add timestamps to the Console Output  
☐ Use secret text(s) or file(s)

### Build

Execute shell

Command 
  
See [the list of available environment variables](#)

빌드를 위해 maven 을 실행하였다. -Dmaven.test.skip=true 는 test 를 실행하지 말라는

옵션이다. 필요에 따라서 true 를 false 로 설정하도록 한다.

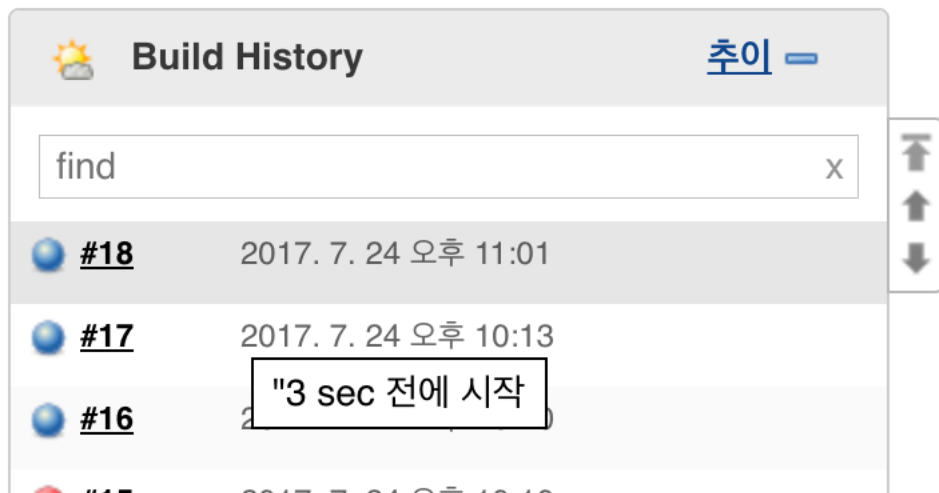
앞에서 작성한 deploy.sh 를 실행하도록 하였다.

cp /home/carami/jenkins/workspace/todo/target 에서 jenkins 는 빌드를 수행한다.

The screenshot shows the Jenkins web interface for a project named 'todo'. At the top, there's a header with the Jenkins logo and the project name. Below the header, there's a sidebar with various icons and labels: '대시보드로 돌아가기' (Return to dashboard), '상태' (Status), '변경사항' (Changes), '작업공간' (Workspace), 'Build Now' (Build Now), 'Project 삭제' (Delete Project), and '구성' (Configure). The main area is titled 'Project todo' and contains a '작업 공간' (Workspace) section with a '최근 변경사항' (Recent changes) link. Below this, there's a '고정링크' (Fixed links) section with a list of links: 'Last build. (#17).46 min 전', 'Last stable build. (#17).46 min 전', 'Last successful build. (#17).46 min 전', 'Last failed build. (#15).50 min 전', 'Last unsuccessful build. (#15).50 min 전', and 'Last completed build. (#17).46 min 전'. On the left, there's a 'Build History' table with a search bar and a list of builds. The table has columns for build number, status, and time. The builds listed are #17, #16, #15, #14, #13, and #12, all with a status of 'Success' and a time of '2017. 7. 24 오후 10:13'.

Build Number	Status	Time
#17	Success	2017. 7. 24 오후 10:13
#16	Success	2017. 7. 24 오후 10:10
#15	Failure	2017. 7. 24 오후 10:10
#14	Success	2017. 7. 24 오후 10:08
#13	Success	2017. 7. 24 오후 10:06
#12	Success	2017. 7. 24 오후 10:06

위의 화면은 17 번 빌드를 수행한 화면이다. 빌드를 수행하고 싶다면 "Build Now"버튼을 클릭한다.



빌드를 수행하면 위와 같이 #18 이라는 새로운 빌드가 수행된 것을 알 수 있다. #18 이라는 링크를 클릭해보자.



위와 같이 어떤 브랜치의 내용이 빌드되었는지 확인된다. Console Output 을 클릭한다.

```

[INFO] Tests are skipped.
[INFO]
[INFO] --- maven-war-plugin:2.2:war (default-war) @ todo ---
[INFO] Packaging webapp
[INFO] Assembling webapp [todo] in [/home/carami/.jenkins/workspace/todo/target/todo-0.0.1-SNAPSHOT]
[INFO] Processing war project
[INFO] Copying webapp resources [/home/carami/.jenkins/workspace/todo/src/main/webapp]
[INFO] Webapp assembled in [67 msecs]
[INFO] Building war: /home/carami/.jenkins/workspace/todo/target/todo-0.0.1-SNAPSHOT.war
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 3.079 s
[INFO] Finished at: 2017-07-24T23:01:05+09:00
[INFO] Final Memory: 21M/172M
[INFO] -----
+ /apps/tomcat/deploy.sh
Killing 9527 ...

Tomcat is being shutdown.
Tomcat started.
Finished: SUCCESS

```

빌드 과정이 출력된 것을 알 있다. 하단에 보면 9527 pid 를 가지고 있는 프로세스가 종료되었고, Tomcat 이 다시 시작된 것을 알 수 있다. 소스코드를 수정하고 git 에 push 한 후 빌드를 수행하여 제대로 반영되는지 확인해 본다. <http://ip> 로 확인